WHAT IS CLAIMED IS:

1. A method for managing stacks of a multitask system comprising the steps of:

storing an internal information of CPU in a task stack of a task to be interrupted as a first area in accordance with the generation of an interruption;

storing a value of a stack pointer after storing the internal information in a prescribed first position of an interrupt processing stack;

setting the stack pointer to a prescribed second position of the interrupt processing stack; and

starting an interrupt process,

wherein the prescribed second position of the interrupt processing stack corresponds to a prescribed position in the first area in the task stack of a specific task.

2. A method for managing stacks according to claim 1, wherein the first area includes a second area for storing task control information necessary for a return control from an interrupt process and a third area disposed immediately after the second area to store the internal information of the CPU except the task control information, and

the prescribed position in the first area in the task stack of the specific task corresponds to the top address of the third area in the first area.

3. A method for managing stacks according to claim 1, wherein the prescribed first position of the interrupt processing stack is the top address of the interrupt processing stack.

4. A method for managing stacks according to claim 1, wherein the prescribed second position of the interrupt processing stack is located immediately after the prescribed first position of the interrupt processing stack.

5. A method for managing stacks according to claim 1, wherein the first area includes a second area for storing task control information necessary for a return control from an interrupt process and a third area disposed immediately after the second area to store the internal information of the CPU except the task control information, and

the prescribed position in the first area in the task stack of the specific task is an address obtained by correcting the top address of the third area in the first area by a prescribed address difference value.

6. A method for managing stacks according to claim 5, wherein the address difference value is given to designate an area necessary for storing the internal information of the CPU used in the specific task.

7. A method for managing stacks according to claim 5, wherein the address difference value is previously given as a constant.

8. A method for managing stacks according to claim 5, wherein the address difference value is set by executing the specific task.

9. A method for managing stacks according to claim 2, wherein the task control information includes at least a program counter and a program status word (PSW) indicating the state of the CPU.

10. A method for managing stacks according to claim 1, wherein the specific task is an idle task which loops itself without using the internal information of the CPU.

11. A method for managing stacks according to claim 1, wherein the specific task is a task for controlling a shift and a return to a low power consumption mode.

12. A method for managing stacks according to claim 2, wherein no information is stored in the second area, the task control information and the internal information of the CPU

located within a prescribed range are not stored in the task stack of the task to be interrupted but stored in a memory area different from the task stack and the internal information of the CPU except the internal information of the CPU located within the prescribed range is stored in the third area.

13 A stack controller of a multitask system in which information in a CPU is stored in the task stack of a task to be interrupted in accordance with the generation of an interrupt to have a first area, the value of a stack pointer after the storage in the first area is stored in a prescribed first position of an interrupt processing stack and the stack pointer is set to a prescribed second position of the interrupt processing stack to start an interrupt process, wherein a control mechanism is provided by which the top address of the interrupt processing stack is allowed to correspond to a prescribed position in the first area in the task stack of a specific task.

14. A stack controller according to claim 13, wherein the first area includes a second area for storing task control information necessary for a return control from an interrupt process and a third area disposed immediately after the second area to store the internal information of the CPU except the task control information, and the prescribed position in the first area in the task stack of the specific task corresponds

to the first address of the third area in the first area.

15. A stack controller according to Claim 13, wherein the first area includes a second area for storing task control information necessary for a return control from an interrupt process and a third area disposed immediately after the second area to store the internal information of the CPU except the task control information, and the prescribed position in the first area in the task stack of the specific task is an address obtained by correcting the first address of the third area in the first area by a prescribed address difference value.

16. A stack controller according to claim 15, wherein the address difference value is held in an address difference value storing unit showing an area necessary for storing the CPU information used in the specific task.

17. A stack controller according to claim 15, wherein the address difference value is previously given as a constant.

18. A stack controller according to claim 15, wherein the address difference value is set by executing the specific task.

19. A compiler for automatically forming the address

difference value in the method for managing stacks according to any one of claims 5 to 8 or in the stack controller according to any one of claims 15 to 18.